Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 59 (2015) 14 – 18

Procedia Computer Science

International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

A Web-Based Chinese Chess Xiang Qi using *n*-tier Architecture Model

Novita Hanafiah, Jaka Hartanto, Yulyani Arifin*, Hengky Frans, and Winardi Cristian

Bina Nusantara University, Jl. K.H Syahdan No.9, DKI Jakarta, 11480, Indonesia

Abstract

Classic game has its attraction so that it is not forgotten easily along the time, for instance Chinese chess called Xiang Qi. The current technology development makes the classic games have to adapt by converting the style of play, such as online version of the classic game. *N*-tier architecture provides a model for software developers to create an application which has a high flexibility and reusability. By dividing the application into some tiers, developers have options to modify or add a particular layer, instead of changing the whole application. The implementation of *n*-tier can be done by supporting with internet. The combination of Xiang Qi with online game technology based on web is expected to be successful in attracting the attention of gamers/players. The paper presents a re-packaging of conventional games into an online game application which built using *n*-tier developing approach that can manage accessibility and interactivity of the games.

Peer-review under responsibility of organizing committee of the International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

Keywords: ID card; character; recognition; OCR; template; matching;

1. Introduction

In the last decades, Internet be a mainstay in distributing information and communication, which scope various aspects such as business, organization, government, education, entertainment, etc. The architecture of those

* Corresponding author. Tel.:+62-812-909-569-3.
E-mail address: yarifin@binus.edu

Peer-review under responsibility of organizing committee of the International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)
doi:10.1016/j.procs.2015.07.332

transaction is called web service. Many web based applications are built, including games. Development of games itself follows the rapid technological developments, where the game can be played online by two or more people at the same time in different places. Classic game can be popularized back by applying the web service technology, so it can continue to be known of all time. Xiang Qi is one kind of chess played in China, it is known as Chinese chess. Regarding to the research in 1990[1], Xiang Qi is evolution of the classic Chinese game, Liubo. Like Liubo, the set chess of Qiang Xi consist of 5 pawns, and the mission is to catch the general.

Some design issue for massively multiplayer online games is introduced by Lu Fan et al.[2], while a hybrid game architecture is presented by Jared and Daniel[3]. The hybrid architecture combines the advantage of client-server and peer-to-peer event distribution to maintain the centralized control state and reducing the server bandwidth, and focus on role-playing. Client-server architecture is simpler to implement, since it does not required peer-discovery, distributed event ordering and cheat prevention. The experimental result shows the hybrid architecture can save considerable bandwidth for the central server. Chang et al.[4] propose methods to quantify online games QoE (Quality of Experience) under various network situations. The paired comparison and probabilistic choice is used to measure the game quality.

There are many types of web application architecture. Akram and Hayat[5] describe some architectural view point in web application and structured of tiered architecture. Each tier architecture have different level of performance, availability and complexity. Focus group is used for qualitative approach to analyze the web application architecture, meanwhile experiment is used in the quantitative approach. The experimental result shows that vertical 3-tier architecture is the best among vertical 2-tier, horizontal 3-tier, and diagonal 3-tier architecture. CEMARA[6] is a web dynamic application which is dedicated to collect continuous and complete records of all patients presenting with a rare disease and their follow-up. The system applies client-server architecture. The information system is based on *n*-tier architecture, where the clients tier connect to the middle tier via web browser and the middle tier is connected to several databases. Web service architecture in 3 tiered system is composed of a front end web server, an application server, and backend database server. Liu et al.[7] present the analytical model of the 3-tiered web service architecture by using queueing network theory to model the architecture. The testbed is built based on the deployed combination of Apache, Tomcat, and MySQL, with the client request generator being the industry standard TPC-W benchmark.

Nevertheless, a web based application has some characteristics which need to be considered, such as availability, concurrency, data-driven, and performance[8]. The architectural design of a web application becomes a challenge so that the web can remain available to be accessed by the client, give a good performance, as well as having a low level of complexity. In addition, the game should include all players who want to play at the same time and prevent bottlenecks. Lu et al.[9] said the basis of performance analysis and capacity planning is a validated model. Regarding to bottleneck issue, the modern servers are typically multithread or multiprocess based, so the M/G/m queue method is proposed to model the web service architecture. The performance metric of interest is the response time of client request. The result indicates the M/G/m queue provides reasonable estimate of the service response time for moderately high traffic intensities.

On the other side, Xu et al.[10] present evaluation mode to evaluate the peak load of web server based on special relationship between response time and the throughput when the request rate is lower than peak load. Model Based on the Response Time (MBRT) provides method to compute the peak load under a given server configuration. The peak load is measured accurately by Controlled Markov Model. However, the MBRT is hard to find out the bottleneck of a web server. Cai et al.[11] introduce an approach to predict the average response time as one quality of services (QoS) metric of typical 3-tier web application. Queuing network theory is used to model the application as a network of resources queues.

In 2007, a design of Chinese Chess game has been implemented in Java technology[12]. It applies an algorithm follow the rules of Chinese chess from moving area, moving direction and passing hurdles. This paper presents the game Xiang Qi which is built in the form of web application adopted *n*-tier architecture web design. *N*-tier architecture provides a model for software developers to create an application which has a high flexibility and reusability. By dividing the application into some tiers, developers have options to modify or add a particular layer, instead of changing the whole application. Typically, there are 3 tiers which are presentation tier, domain logic tier and data storage tier. The communication between those layers assisted by the cross-cutting layer. The combination of Xiang Qi with online game technology based on web is expected to be successful in attracting the attention of

gamers/players. The paper presents a re-packaging of classic games into an online game application which built using *n*-tier developing approach that can manage accessibility and interactivity of the games.

2. Xiang Qi Web Base Architecture

Web-based Xiang Qi application architecture is designed in client-server architecture. The system applies the MVC (Model View Controller) concept and created four layers in the architecture which are presentation layer, business layer, data layer and cross-cutting layer. The logical layer/business layer separates the task for each component and the design is easy to reuse. Figure 1 illustrates the overview mechanism of Xiang Qi game. User interacts with the web application and in this step the presentation layer sends information to the cross-cutting layer. Afterwards, that information is passed to the business layer before it goes to the data layer and get the data from data source to be shown to the customer.

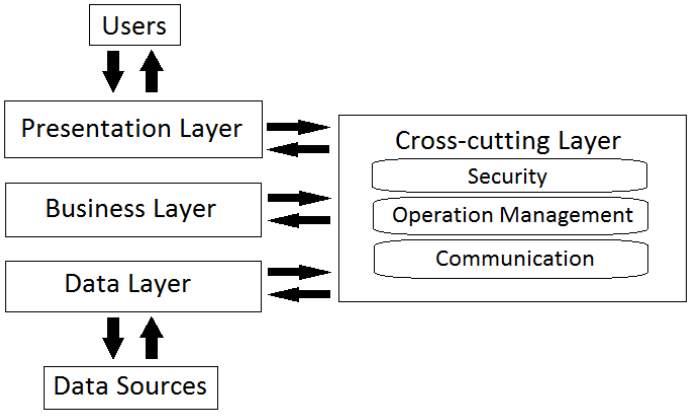

Fig 1. *n*-tier architecture of Xiang Qi game

The essential component in this architecture design is the cross-cutting layer. When a requests from a user is received in the presentation layer, the security and operation management component in the cross-cutting layer is worked and return false or true value. If the user need to register or login, the operation management gives false value, so that the presentation layer can generate a new UI component. On the other side, the true value is returned when a user data is found by the Security object, and the business layer send information for the presentation layer to generate the menu that can be opened by the user based on the operation management object. The same mechanism when users start to play the game. The communication object in the cross-cutting layer is used due to the game rules about switching the player turn.

The number of tier can increase the complexity of the system and effected the deployment effect and cost. Thus, the number of tier needs to be minimized to solve the scalability, security and failover of a system. However, there is no exact number of tiers needed to develop a system because it depends on the requirements of the system to achieve the scalability and performance.

3. Experiment Result

Xiang Qi application has minimum requirements of software specifications on the server side, such as windows server operating system, Apache web server version 6.0, and MySQL DBMS version 5. Meanwhile, the client-side needs windows 7 as the operating system and Mozilla Firefox version 3.6.13 as the web browser. Apache Bench is used in the experiment to measure the performance of web server. It can show how many request of each second our web site can serve the user by sending an arbitrary number of request. In addition, Apache Bench supports concurrency. The test is done in Intel (r) Core(TM) i5 CPU 760 @2.80GHz with 4GB RAM, and windows 8 64-bit operating system. The performance result is shown in the table 1. Number of request used for the test is started from 1 and increased up to 300,000. We experiment it with 20,000 of concurrency level for each request, so the total

request faced by the server is 300,000 x 20,000 or as much as 6,000,000 requests. All of the request is completed processed and there is no failed request. Time taken for the test is increased linearly with increasing number of request.

Table 1 presents the information that *n*-tier architecture can handle number of requests efficiently. It is proven by the transfer rate is fairly stable for test number 2 to 12. Request per second can also be coped well with the completion of approximately 1-2 requests per second. Time per request needed to complete each request is relatively stable where the average number is in range of 11-16 milliseconds. *N*-tier architecture divides the file into more focused. Since this architecture design is able to address three major problems on the application such as user interface, application logic, and connection to the database, thus, time required to complete each request is relatively stable, which can be seen from the table (across all concurrent request).

Table 1. The capability of Xiang Qi web server

No.	number of request	time taken for test (second)	total transferred (byte)	request per second (mean)	time per request		transfer rate (Kbytes/second)
					mean (ms)	across all concurrent request (ms)	
1	1	7.55	161,440,000	**2,647.42**	0.378	**0.378**	20,869.1
2	20,000	13.55	161,440,000	1,476.14	13,548.85	0.677	11,636.14
3	30,000	17.72	242,160,000	1,693.65	11,808.81	0.59	13,350.73
4	40,000	27.54	322,880,000	1,452.35	13,770.79	0.689	11,448.6
5	50,000	34.64	403,600,000	1,443.38	13,850.33	0.693	11,377.93
6	60,000	44.9	484,320,000	1,336.3	14,966.66	0.748	10,533.83
7	70,000	50.9	565,040,000	1,375.13	14,544.08	0.727	10,839.89
8	80,000	61.75	645,760,000	1,295.59	15,436.99	0.772	10,212.89
9	90,000	64.39	726,480,000	1,397.79	14,308.25	0.715	11,018.55
10	100,000	78.8	807,200,000	1,269.12	15,758.94	0.778	10,004.24
11	200,000	164.7	1,614,400,000	1,214.36	16,469.6	0.823	9,572.56
12	**300,000**	**202.9**	2,421,600,000	1,478.46	13,527.55	0.676	11,654.46

Each architecture certainly has limitations in dealing with the request. Likewise with *n*-tier Architecture, with the increasing number of requests made *n*-tier architecture is becoming increasingly severe in completing the request. Server need to be restarted because it cannot response the load of 400,000 to 800,000 requests. Table 2 shows the experiment result after the server is restarted. The test starts with 900,000 requests and the server does not stop working but refused the connection after 10 failure tests. Xiang Qi application itself has 20 Megabytes size, so the server cannot handle all of the requests in table 2 with only 1 server 4GB RAM. In the test number 12 on table 1, the number of request is 300,000 with 20,000 concurrency, which is about 300,000 x 20,000 = 6,000,000 request. With the size of every request is 20,000,000 bytes, then the actual request made by the user to the server is equal to 6,000,000 x 20,000,000 bytes = 120,000,000,000,000 bytes or 120Terrabytes. This amount exceeded the size of RAM server that is used to test. It shows that *n*-tier architecture is able to optimize the use of server up to 30,000 times or 300%.

Table 2. The maximum capability of Xiang Qi web server

No.	number of request	time taken for test (second)	complete request	failed request	total transferred (byte)	request per second (mean)	time per request		transfer rate (Kbytes/second)
							mean (ms)	across all concurrent request (ms)	
1	900,000	-	36,355	-	-	-	-	-	-
2	1,000,000	-	395,286	-	-	-	-	-	-
3	5,000,000	20.92	12,980	6	161,440,000	620.600	32226.95	1.611	7537.85
4	8,000,000	-	1,196,477	-	-	-	-	-	-
5	9,000,000	-	80,201	-	-	-	-	-	-
6	10,000,000	73.08	81,368	1	807,200,000	1,113.520	17,960.33	0.898	10,786.86
7	40,000,000	14.08	13,023	-	161,440,000	925.280	21,613.51	1.081	11,197.08
8	50,000,000	346.83	455,869	1	3,713,120,000	1,314.41	15,215.87	0.761	10,455.06

4. Conclusion

Implementation of *n*-Tier architecture on the application of Xiang Qi game is simple and reusable. By using the cross-cutting layer, the security level is higher than typically client-server model and the communication between presentation, business and data layers is managed by the operation management. The task for each component is defined clearly in the business layer. The presentation layer generates a web application that is intuitive, interactive, with an interesting look as well as suitable to the characteristics of Xiang Qi players in General. The experimental result shows a good performance, which can achieve about 1 second to process each request across the entire concurrent request. However, the system in Xiang Qi game still needs some improvements to make the web applications are becoming more dynamic. Some other challenges in game also need to be considered, such as cheating issue. Moreover, creating a *n*-tier architecture need to analyze the advantage and disadvantage to get the best solution of the system Developer can choose the main focus in implementing a solution between a number of users (indicate by number of request), the speed of application, or response time. Developers can also choose a main focus in implementing a solution between the number of users indicated by the number of requests, the speed of application shown in Time taken for a test or response time indicated by the time per request and the request per second.

References

1. (Yutopian Enterprises, 1999, History of Xiang Qi).
2. Fan, Lu, Phil Trinder, and Hamish Taylor. Design issues for peer-to-peer massively multiplayer online games. International Journal of Advanced Media and Communication. 4(2); 2010: 108-125.
3. Jardine, Jared, and Daniel Zappala. A hybrid architecture for massively multiplayer online games. Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games. ACM; 2008.
4. Chang, Yu-Chun, et al. Online game QoE evaluation using paired comparisons. Proceedings of IEEE CQR 2010; 2010: 1-6.
5. Akram, Muhammad, and Sikandar Hayat+. Qualitative & Quantitative analysis of tiered Architecture of Web-Applications.
6. Messiaen, Claude, et al. CEMARA: a Web dynamic application within a N-tier architecture for rare diseases. Studies in health technology and informatics. 136; 2008: 51.
7. Liu, Xue, Jin Heo, and Lui Sha. "Modeling 3-tiered web applications." Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on. IEEE; 2005.
8. Pressman, Roger S. Software Engineering: A. Practitioner Approach. Seventh Edition. McGraw-Hill Companies, Inc., New York; 2010
9. Lu, Jijun, and Swapna S. Gokhale. Performance Analysis of Web Service Architecture.
10. Xu, Xianghua, et al. "Performance evaluation model of Web servers based on response time." Conference Anthology, IEEE. IEEE, 2013.
11. Cai, Duan, Qixun Zhang, and Zhonghai Wu. A Feasible Approach to Predict the Response Time for 3-Tier Web Application with Service Demand Law. Recent Advances in Computer Science and Information Engineering. Springer Berlin Heidelberg; 2012.p. 183-193.
12. Hao, G. A. O. W. G. U. O. J. The Design and Implementation of Chinese Chess Game Based on JAVA. Journal of Dalian Nationalities University. 5; 2007: 008.